Intrusion Alert Correlation to Support Security Management

Cláudio Toshio Kawakani
State University of Londrina
Computer Science Department
Londrina, PR, Brazil
claudio.tk93@gmail.com

Sylvio Barbon Junior
State University of Londrina
Computer Science Department
Londrina, PR, Brazil
barbon@uel.br

Rodrigo Sanches Miani
Federal University of Uberlândia
School of Computer Science (FACOM)
Uberlândia, MG, Brazil
miani@ufu.br

Michel Cukier
University of Maryland
A. James Clark School of Engineering
College Park, MD, USA
mcukier@umd.edu

Bruno Bogaz Zarpelão
State University of Londrina
Computer Science Department
Londrina, PR, Brazil
brunozarpelao@uel.br

ABSTRACT

To support information security, organizations deploy Intrusion Detection Systems (IDS) that monitor information systems and networks, generating alerts for every suspicious behavior. However, the huge amount of alerts that an IDS triggers and their low-level representation make the alerts analysis a challenging task. In this paper, we propose a new approach based on hierarchical clustering that supports intrusion alert analysis in two main steps. First, it correlates historical alerts to identify the most typical strategies attackers have used. Then, it associates upcoming alerts in real time according to the strategies discovered in the first step. The experiments were performed using a real data set from the University of Maryland. The results show that the proposed approach can provide useful information for security administrators and may reduce the time between a security event and the response.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; K.6.m [**Management of Computing and Information Systems**]: Miscellaneous—*Security*

General Terms

Security, Management

Keywords

Intrusion Detection, Alert Correlation, Security Management, Data Mining

SBSI 2016, May 17th-20th, 2016, Florianópolis, Santa Catarina, Brazil

1. INTRODUCTION

Information and Communication Technology (ICT) plays a significant role in modern organizations, bringing many benefits concerning facility and productivity. Information is a valuable asset to every organization and its value dramatically increased in recent years [10]. Therefore, a security incident may cause loss of productivity, resources and reputation [1]. Incident management performs a relevant task for the information security of an organization, which defines processes to detect, analyze and respond to an incident [9].

Network or system intrusions can cause security incidents. Intrusion Detection Systems (IDS) are devices used to monitor information systems and networks for these intrusions. When the IDS detects a sign of security violation, it generates an alert and saves them into a log file. A better understanding of intrusions is possible by the analysis of alerts that the IDS triggers. Thereby, a security administrator can have a better situational awareness of the intrusions and efficiently respond to it. However, due to the huge amount of alerts that the IDS generates, the manual analysis of the IDS alerts becomes a time consuming and error prone task [4].

The time between an incident and its response must be reduced to limit the damage and lower the cost of recovery [9]. Data mining techniques can be used to cut this time, since it supports the analysis of a huge amount of intrusion alerts by finding interesting patterns and summarizing them into improved information structures [4]. In this work, we propose a new approach to assist the security analyst in the intrusion alert analysis. The proposed approach is composed of two elements: the offline correlator and the online correlator.

The offline correlator aggregates historical alerts using the connected components method, extracts attack strategy graphs and uses hierarchical clustering to group similar attack strategy graphs. For this, a new method to compare the similarity between two attack strategy graphs is also pro-

posed. The attack characteristics of each generated cluster are then identified.

The online correlator generates hyper-alerts which contain useful attributes that assist the security analyst. A hyper-alert is composed by different low-level alerts and it is updated in real time as the upcoming low-level alerts are triggered. Also, each hyper-alert is associated to a cluster that the offline correlator defined previously. By analyzing the cluster information, the security analyst can better understand the characteristics of an attack when only the first alerts are available. The experiments were performed using a real data set gathered by an IDS device deployed in the University of Maryland with about 40,000 computers.

The rest of this paper is organized as follows: Section 2 presents the related work, Section 3 explains the offline and online correlators, Section 4 discusses the experiments and analyzes the results. Section 5 concludes this paper.

2. RELATED WORK

Researchers have proposed different methods to assist intrusion alert analysis in recent years. The proposed methods aim to reduce the amount of alerts, discover relationships between them and provide a visual representation for the correlated alerts.

Because of the enormous amount of alerts that IDSs trigger, Julisch [4] proposes an approach to handle the IDS alerts more efficiently. The proposed approach is based on the root-cause detection. A root-cause is the reason for the occurrence of an alert. Julisch claims that few root causes are responsible for 90% of the alerts. A clustering technique is proposed to group the alerts, supporting the root-cause analysis. The author evaluated the proposed approach with real data that was collected from a commercially used network.

To improve the quality of intrusion alerts information, Spathoulas and Katsikas [12] propose a system to process the intrusion alerts that belong to multiple IDSs and provide a high-level presentation of the security events. For this, the alerts that have the same source and destination IP addresses, same attack identifier and are close in time are grouped. Then, the similar groups are clustered, and for each cluster, a visual representation is provided. The proposed system was evaluated using the DARPA 2000 data set and an academic data set generated by attack simulation.

Considering that one attacker performs multiple attack steps to reach its goal, Xuewei et al. [16] propose an approach to automatically identify the multistage attacks in intrusion alerts. First, alerts are clustered by related IP addresses using the method of connected components. Then, for each cluster, a directed graph that represents the attacker steps is generated. Each node in the graph represents an attack type, and each edge contains the probability of an attack type succeeds another attack type. The proposed approach was evaluated with the DARPA 2000 data set. Results showed that it could uncover the attack scenarios properly.

Ghasemigol and Ghaemi-Bafghi [2] propose a system that correlates intrusion alerts using their entropy. The goal of the proposed system is to find causal relationships between the intrusion alerts, providing a high-level representation of the attack scenario. First, the number of alerts is reduced using data aggregation. Data aggregation is applied to group the alerts in clusters and the graphs of the clusters are generated for visualization. The authors reached a rate of 99.98% of data reduction using the DARPA 2000 data set to evaluate the proposed system.

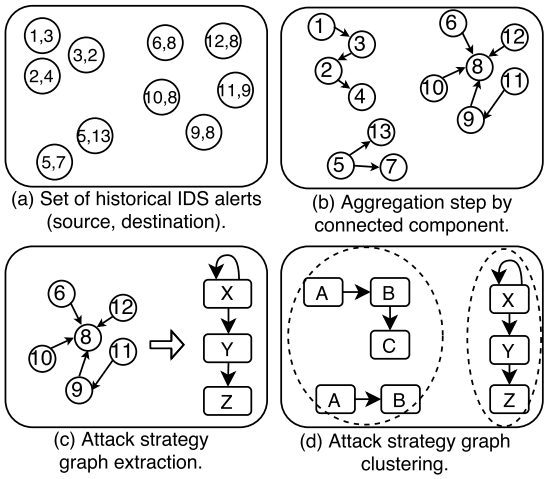

Figure 1: Offline correlator overview.

Shittu et al. [11] propose a framework for intrusion alert analysis. The proposed framework reduces the quantity of false positive alerts and uses post-correlation methods to support the alert analysis. For this, a metric for prioritizing alerts and a method to cluster the similar alerts using DBSCAN were presented. Graph Edit Distance is used to compute the difference between the correlated alert graphs. The data set used to test the framework belongs to the 2012 cyber range experiment that industrial partners of the British Telecom Security Practice Team performed.

In our work, we propose a model that generates hyper-alerts online considering the historical behavior of the attackers. This model is based on the agglomerative hierarchical clustering using Ward's method, which groups attack strategies according to a new metric we propose to compare their similarity. The proposed model also takes into account the necessary time to notify the security analyst with high-level information of the event. It is important to emphasize that a real data set was used to evaluate the proposed model, unlike the most of the related work, which use data sets from the DARPA challenge.

3. PROPOSED APPROACH

The proposed approach is composed of two correlators: the offline correlator and the online correlator. The offline correlator receives as input a set of IDS alerts and constructs the cluster model of attack strategies to be used in the online correlation. The online correlator analyzes the incoming alerts in real time and extracts useful information by matching these alerts with the cluster model that the offline correlator constructed. These two components are described in the next subsections.

3.1 Offline Correlator

The offline correlator aims to get a set of historical IDS alerts and organize them into clusters, giving a better understanding of the attacks. For this, the first step consists in separating the alerts into connected components (aggrega-

Table 1: First example of IDS alerts.

Time	Source	Destination	Signature
01/01/2016 00:30:00	10.0.0.0	10.0.0.1	A
01/01/2016 00:35:00	10.0.0.1	10.0.0.2	B
01/01/2016 00:40:00	10.0.0.1	10.0.0.3	C
01/01/2016 00:45:00	10.0.0.64	10.0.0.65	D
01/01/2016 00:50:00	10.0.0.64	10.0.0.65	D
01/01/2016 00:55:00	10.0.0.64	10.0.0.65	E
01/01/2016 01:00:00	10.0.0.64	10.0.0.65	F

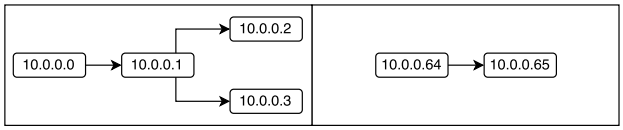

Figure 2: Connected components for the first example of IDS alerts.

tion step). Then, for each connected component, the attack strategy graph is extracted. After that, the attack strategy graphs are separated into clusters using hierarchical clustering techniques (clustering step). Figure 1 presents an overview of the offline correlator. This process is detailed next.

Let s, d, t, k be the source IP address, the destination IP address, the timestamp and the attack signature of an alert, respectively. Then, an alert A is defined as the 4-tuple $A = \langle s, d, t, k \rangle$. The offline correlator uses the IP address and timestamp information to organize the alerts. This organization is based on the method that Treinen and Thurimella [13] used in their work, known as connected components. In the aggregation step, the connected component organization separates the alerts by their IP addresses. If the alerts have related IP addresses, i.e. $A_1.s = A_2.s$ or $A_1.d = A_2.d$ or $A_1.s = A_2.d$ or $A_1.d = A_2.s$, then they are grouped. In this work, another condition must be satisfied to group the alerts: the alerts must have a difference of timestamps shorter than x minutes, i.e. $A_2.t - A_1.t < x$ minutes.

For example, considering the set of alerts of Table 1, two connected components can be constructed. Figure 2 shows the connected components, where each node represents the IP address, and each directed edge represents the direction of the attack (from source to destination).

Another example, from Table 2, highlights the importance of the condition related to the difference of timestamps of alerts with related IP addresses. All of the alerts in Table 2 have related IP addresses, but three of them were recorded hours after the first ones. Considering a maximum time of 60 minutes for the difference of timestamps of alerts with

Table 2: Second example of IDS alerts.

Time	Source	Destination	Signature
01/01/2016 00:00:00	10.0.0.0	10.0.0.1	A
01/01/2016 00:40:00	10.0.0.1	10.0.0.2	B
01/01/2016 01:20:00	10.0.0.1	10.0.0.3	B
01/01/2016 10:00:00	10.0.0.0	10.0.0.4	A
01/01/2016 10:40:00	10.0.0.4	10.0.0.5	B
01/01/2016 11:20:00	10.0.0.4	10.0.0.6	B

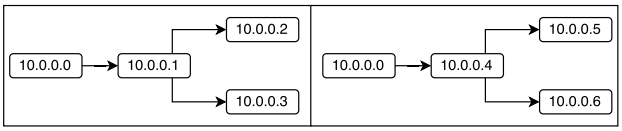

Figure 3: Connected components for the second example of IDS alerts.

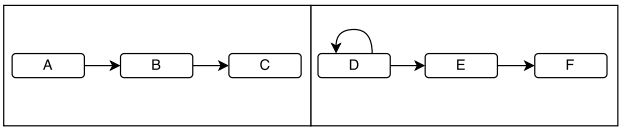

Figure 4: Attack strategy graphs of the alerts in Table 1.

related IP addresses, the first three alerts compose one connected component, and the last three alerts compose another connected component, illustrated in Figure 3.

Using the alerts signature and timestamp makes it possible to determine the sequence of signatures triggered in each connected component. The sequence of signatures is represented in a directed graph format, where each node represents one signature and each edge represents the sequential relationship between the signatures. This graph is named attack strategy graph. One attack strategy graph is generated for each connected component. Figure 4 shows the attack strategy graphs created for each connected component of the alerts from Table 1.

The next step of the offline correlator is to group similar attack strategy graphs. For this, a method to verify if two graphs are similar (or dissimilar) should be defined. Ning and Xu [7] verified the similarity between two attack strategy graphs using the Graph Edit Distance technique. The Graph Edit Distance technique measures how many edit operations (node insertion, deletion or substitution, and edge insertion, deletion or substitution) are required to transform one graph into another. The more edit operations are needed, more dissimilar are two graphs. However, this method showed to be inadequate to our attack strategy graphs due to its performance when applied to a real data set. Also, for smaller graphs, the Graph Edit Distance is low even if the graphs are completely different, because few edit operations are necessary. Figure 5 demonstrates this problem: the cost for both situations will be one node substitution (X for Y), even taking into account that the situation (a) shows similar nodes (A and B). Therefore, if two graphs have a huge amount of similar edges and nodes and only one different node, the Graph Edit Distance technique will consider only the different node.

When the Graph Edit Distance is applied to compare attack strategy graphs, each edit operation must have a predefined cost. As Ning and Xu [7] stated, each stage of the attack (each node of the attack strategy graph) is more significant than the causal relationship between the attacks (the edges of the strategy graph). Therefore, the cost of node operations, in Ning and Xu work, was proposed to be significantly higher than the cost of edges operations. Following this reasoning, we propose a new method to measure attack graph similarity using Jaccard index. Since, according to Ning and Xu, the nodes of an attack strategy graph are

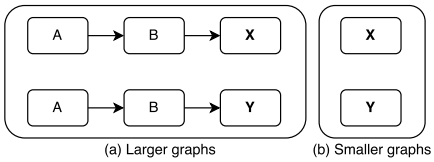

Figure 5: Example of attack strategy graphs for Graph Edit Distance and for the new method based on Jaccard index.

more significant than the causal relationship between them, the use of an algorithm like Graph Edit Distance showed to be unnecessary.

Jaccard index measures the similarity between two finite sets. Considering M and N two finite sets, the Jaccard index between M and N is defined as the size of the intersection of M and N, divided by the size of the union of M and N (Equation 1) [8]. If both sets are empty, the Jaccard index is defined as 1. To measure the similarity between two attack strategy graphs, two Jaccard indexes were calculated, first for all nodes, then for all pair of nodes (representing the edges). Then, the mean of the two results was considered the similarity between two attack strategy graphs. For the attack strategy graphs of the situation (a) in Figure 5, the similarity is calculated as follows:

1. First, it is calculated the similarity S_1 of nodes, which is the size of the intersection of the nodes, i.e. $\{A, B\}$, divided by the size of the union of nodes, i.e. $\{A, B, X, Y\}$. Therefore, $S_1 = 2/4 = 0.5$.

2. Then, it is calculated the similarity S_2 of the edges, which is the size of the intersection of the edges, i.e. $\{A \rightarrow B\}$, divided by the size of the union of edges, i.e. $\{A \rightarrow B, B \rightarrow X, B \rightarrow Y\}$. Therefore, $S_2 = 1/3 = 0.3$.

3. The final similarity is the mean between S_1 and S_2, which is 0.4.

$$J(M, N) = \frac{|M \cap N|}{|M \cup N|} \quad (1)$$

With this new metric, the problem presented in Figure 5 is solved, because all nodes and their edges are taken into consideration. Also, this metric does not present performance problems, as observed with the Graph Edit Distance.

Using this new metric, all attack strategy graphs can be compared, generating a similarity matrix. The similarity matrix is used as input to the agglomerative hierarchical clustering using Ward's method [14, 6]. Considering z_{ik} the set of elements of a cluster k (which have n_k elements), and considering $\bar{z}_k$ the mean of the elements, the error sum of squares for cluster k is given by the Equation 2.

$$E_k = \sum_{i=1}^{n_k} \parallel z_{ik} - \bar{z}_k \parallel^2 \quad (2)$$

Considering E_k defined by the Equation 2, the Ward's method calculates the total error sum of squares E for all g clusters using the Equation 3.

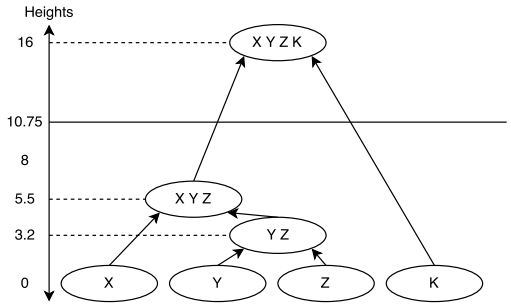

Figure 6: Example of dendrogram for agglomerative hierarchical clustering using Ward's method.

$$E = \sum_{k=1}^{g} E_k \quad (3)$$

The agglomerative hierarchical clustering method begins with n clusters that contain a single element (where n is the number of elements). Then it performs $n-1$ merging steps. At each step, the two clusters that give the smallest increase in the total error of the clusters are merged. The method ends with one cluster that contains all n elements. As a result, this method outputs a dendrogram, which is a tree that represents the entire clustering process (Figure 6). The cutting height of the dendrogram that defines the number of clusters is problem dependent [3, 15]. In this work, the cutting height is defined as the value of the third quartile of the sorted set of heights, which separates the 75% lower heights from the 25% higher heights [5].

Figure 6 illustrates an example of the agglomerative hierarchical clustering technique, where each height represents the square root of the total error of the clusters after a merge between two clusters (calculated with Ward's method) [6]. In this example, the number of elements is four. Therefore, three $(n-1)$ merging steps were required. The first merging step groups the clusters Y and Z, creating the new cluster YZ. The second merging step groups the clusters X and YZ, creating the new cluster XYZ. The last merging step groups the clusters XYZ and K, creating the new cluster $XYZK$. The dendrogram has been cut in the height 10.75 (third quartile of the heights 3.2, 5.5 and 16). Thus, two clusters were generated: the first cluster is composed of the elements X, Y and Z and the second cluster is composed of the element K. In our work, each element is an attack strategy.

After the offline correlation, we have all historical alerts separated into clusters according to the relationships between attack steps. Each cluster groups similar attack strategies and provides a summarized overview of them. The clusters information can be used to support the analysis of the upcoming alerts by the online correlator.

3.2 Online Correlator

The online correlator (Figure 7) receives alerts in real time, correlates them according to the clusters of attack strategies generated in the previous step and creates or updates hyper-alerts. Each hyper-alert is a structure that contains a set of related low-level alerts (representing one con-

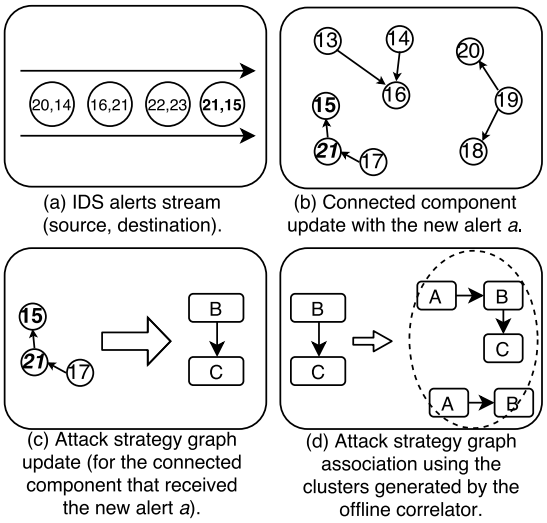

Figure 7: Online correlator overview.

nected component), one attack strategy graph and other useful information. Table 3 describes all attributes of a hyper-alert. The first column represents the category of the attributes, the second column represents the attributes of each category and the third column describes each category. These attributes assist the security management by giving a better insight of what is happening in the network.

To create the hyper-alerts, the online correlator separates the input alerts in connected components, generates the attack strategy graph for each connected component and extracts the other useful information. The techniques used for the connected component identification and attack strategy graph comparison are the same as employed in the offline correlation. The difference from the offline correlator is the alerts are processed as they are generated in real time. Therefore, for each new alert A, it is verified if A belongs to an existing connected component comparing the IP addresses and timestamp (otherwise, A will belong to a new connected component). After determining to which connected component the new alert belongs, the attack strategy graph of this connected component is updated with the new alert A. Then, using the Ward's method, the online correlator associates the updated attack strategy graph G to one of the clusters that the offline correlator generated. Associating the new attack strategy graph to a cluster provides useful information for the security analyst, since each cluster contains information about typical attack strategies that attackers have adopted against the network.

As a result, the online correlator outputs useful information for each hyper-alert (Table 3). Also, the hyper-alert information is updated in real time as it receives a new low-level alert. With this new information, the security analyst has a better situational awareness of the environment in real time to react faster to a security event.

4. EXPERIMENTS AND RESULTS

The IDS alerts used in the experiments were recorded in July 2012 by an IDS deployed in a network with about 40,000 computers from the University of Maryland. The

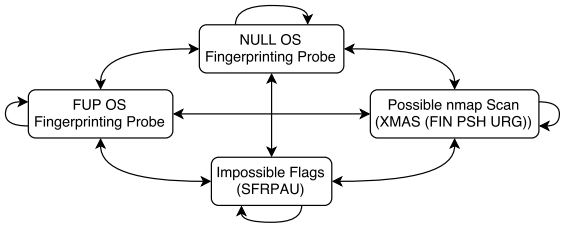

Figure 8: Attack strategy graphs of the Cluster 7.

alerts triggered in the first 14 days of July 2012 were the input of the offline correlator to generate the cluster model of attack strategies. The alerts triggered in the other days of the month (from 15th to 31st of July 2012) were used to test the online correlator. The maximum time difference between two alerts of a connected component was defined as 60 minutes for the offline and online correlator.

For the offline correlator, a total of 20,509 alerts were generated. The aggregation step separated these 20,509 alerts into 895 connected components. Then, the connected components that represent exceptional situations were filtered. A connected component was considered as an exceptional situation if it contains only one alert or contains only alerts with the same signature. These cases do not depict attack strategies used by attackers, as they show an attack-flow with a single step and do not provide enough information on the behavior of the attacker. With this filtering, 3,796 alerts were removed, with 16,713 remaining alerts. These 16,713 alerts are represented by 57 connected components. For each one of these 57 connected components, the attack strategy graph was generated, and the similarity matrix was computed using the proposed metric based on Jaccard index. Finally, the 57 attack strategy graphs were separated into clusters by the hierarchical clustering technique.

A total of 12 clusters were generated. The two clusters (Cluster 1 and Cluster 7) that represent the majority of the alerts were chosen to be further explained in this experiment. Cluster 1 covers 14,138 alerts (84.59% of the 16,713 alerts) and has 10 connected components. Cluster 7 covers 2,317 alerts (13.86% of the 16,713 alerts) and has 12 connected components. This means that 98.46% of the 16,713 alerts are covered by these two clusters. Cluster 1 is characterized by buffer overflow attacks. Figure 8 shows the attack strategy graph of Cluster 7. This cluster is characterized by reconnaissance attacks, such as the port scan Possible Nmap Scan(XMAS (FIN PSH URG)), and the Fingerprinting Probe, which tries to ascertain the operational system of the target.

The alerts triggered From 15th to 31st of July 2012 were used for the online correlator validation. In this period, 19,933 alerts were generated. To test the online correlation, a simulator was developed to release the alerts one by one. As explained in Section 3, for each new alert, it is verified to which connected component it belongs. After determining to which connected component the new alert belongs, the attack strategy graph of this connected component is updated with the new alert. Then, the updated attack strategy graph is associated to one of the clusters generated by the offline correlator.

Therefore, this is a process where each connected component (and consequently each attack strategy graph) changes

Table 3: Hyper-alert information.

Attribute Category	**Attributes**	**Description**
Status information	Status.	The connected component is closed if no alert is added in x minutes, otherwise it is under construction. If the connected component is closed, it means that it will not receive more alerts. Then, this is the final state of the hyper-alert.
Cluster information	Best cluster and increased error.	Reports the best cluster for the hyper-alert and the new error of the cluster, in case of the attack strategy graph of this hyper-alert was added to this cluster (using Ward's method). The bigger the error, the greater the difference between the cluster and the attack strategy graph.
Time information	Start time, end time and duration.	Reports the time of the first and the last alert of the hyper-alert and the difference in minutes between these times.
Quantity information	Quantity of alerts, quantity of distinct attackers, quantity of distinct targets and quantity of distinct signatures.	Counts the amount of alerts, attackers, targets and signatures that composes the hyper-alert.
Address information	The amount of distinct source IP addresses and destination IP addresses that are involved in the event.	This information is useful to determine which IP addresses are responsible to trigger more alerts and which IP addresses receive more attacks.
Signature information	The amount of distinct signatures that occurred and the attack strategy graph.	This information is useful to determine which signature happens more often and to understand the causal relationship between these signatures.
Relationship information	The connected component and a list of relationships.	One relationship consists of one source IP address, one destination IP address, and the number of time that these IP addresses appear in the same alert of the connected component. This information is useful to understand the relationship of the IP addresses.
Attack characteristic information	The mean of the targets per attacker and the mean of attackers per target.	This information is useful to understand the characteristic of the attack. For example, if the mean of targets per attacker is high, it may indicate an reconnaissance attack. Or if the mean of attackers per target is high, it may indicate an DDoS attack.

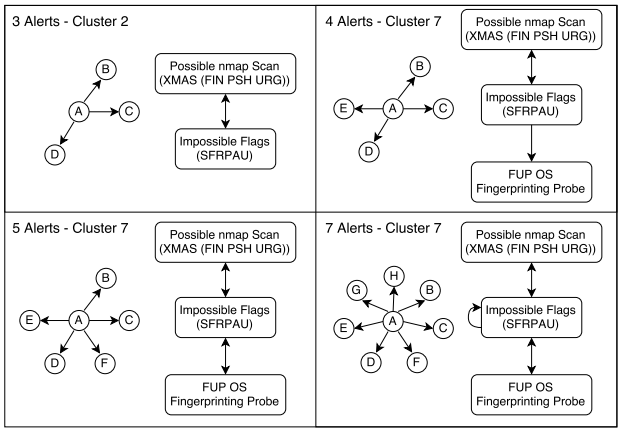

Figure 9: Evolution of one hyper-alert associated to the Cluster 7.

as a new alert is added to it. A new attack strategy graph (updated by only a few alerts) can be associated to a cluster at the beginning and, after receiving a few more alerts, it can be associated to another cluster because the more alerts are used to update the attack strategy graph, the more precise is its association.

After the experiments, the attack strategy graphs showed fast convergence into their final cluster: we observed that an attack strategy graph is correctly associated to its final cluster after an average of 1.54 associations with a standard deviation of 0.7. The average of associations for an attack strategy graph is 8.73 with the standard deviation of 10.05 (the high standard deviation is due to the difference in the attack strategy graphs size). Each new association is performed after the addition of a new alert that changes the attack strategy graph.

A hyper-alert is considered closed when no alerts are added to its attack strategy graph during 60 minutes. After that, no more alerts can be added to this attack strategy graph, and, hence, to this hyper-alert.

Interesting results were observed after analyzing the online correlator output: 16 out of 59 hyper-alerts were associated to the Cluster 1 and contained between 748 and 1209 alerts. By analyzing the time information of the hyper-alerts, we observed that these 16 hyper-alerts were built during the night. Also, each one of the 16 hyper-alerts was generated once a day (from 15th to 31st of July 2012) with the exception of the day 29th. Moreover, by analyzing the address and signature information of the hyper-alerts, we identified that the same source IP address was responsible for about 70% of these alerts, which have the same signature. The security analyst could have identified this pattern in the first days by using the proposed approach. Since this problem is responsible for a significant amount of alerts, the amount of future alerts could have been reduced with the mitigation of this situation.

13 hyper-alerts were associated to the Cluster 7 and have between 6 and 32 alerts. A significant pattern found in each of these hyper-alerts is the presence of a single source IP address attacking multiple destination IP addresses. Figure 9 demonstrates the evolution of a hyper-alert associated to the Cluster 7 by showing its quantity of alerts, best cluster, connected component and attack strategy graph. The IP addresses of the connected components were changed to the labels from A to H for privacy purposes. When the hyper-alert had only three alerts, it was associated to the Cluster 2. After adding a new alert, the association of the hyper-alert changed to Cluster 7 and remained on it until the end (when the hyper-alert was closed with seven alerts). This hyper-alert shows an interesting pattern when its connected component is investigated: only one source IP address is attacking other IP addresses. The mean of targets per attacker is 7, which reinforces the characteristic of a reconnaissance attack. The attack strategy graph of this hyper-alert is very similar to the attack strategy graph of the Cluster 7 (Figure 8). Therefore, with 4 alerts, it was possible to infer that this hyper-alert had the same behavior of other attacks on the Cluster 7.

Thereby, the offline correlator was able to organize the historical intrusion alerts into clusters. These clusters show the summarized information for each attack pattern found in the related alerts and they were useful to support the online correlator. Two main clusters that represent the majority of the alerts (98.46%) were identified, which means that the majority of the attacks directed to this network are similar. The online correlator was able to correlate the alerts in real time, generating the hyper-alerts with much useful information that help the response to security events. Analyzing the hyper-alerts, we were able to find interesting patterns related to the relationship between attacker and target, time, signature and quantity of alerts. Also, we were able to understand the attack scenarios with only their first alerts by analysing the cluster of a given hyper-alert, anticipating the response to a possible intrusion.

5. CONCLUSION

To support the information security management, this paper approached the problem of analyzing and correlating the huge amount of intrusion alerts recorded by an IDS device. A new approach that assists the security analyst in the intrusion alert analysis was proposed. The proposed approach correlates the historical alerts into clusters using data mining techniques and associates the upcoming alerts to these clusters in real time. It was evaluated using a real data set from the University of Maryland. The experimental results show that the proposed approach can discover useful information from historical alerts to assist the analysis of the upcoming alerts. Therefore, the time between a security event and its response may be reduced.

As a future work, the first objective is to evaluate the proposed approach in other data sets, extracted from both real and experimental environments. Moreover, we intend to explore streaming mining techniques, that may help improving the aggregation step of our approach.

6. ACKNOWLEDGMENT

The authors would like to thank Gerry Sneeringer and the Division of Information Technology at the University of Maryland for allowing and supporting the described research.

7. REFERENCES

[1] A. Ahmad, J. Hadgkiss, and A. B. Ruighaver. Incident response teams - challenges in supporting the

organisational security function. *Comput. Secur.*, 31(5):643–652, July 2012.

[2] M. GhasemiGol and A. Ghaemi-Bafghi. E-correlator: an entropy-based alert correlation system. *Security and Communication Networks*, 8(5):822–836, 2015.

[3] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, Sept. 1999.

[4] K. Julisch. Clustering intrusion detection alarms to support root cause analysis. *ACM Trans. Inf. Syst. Secur.*, 6(4):443–471, Nov. 2003.

[5] G. J. Kerns. *Introduction to Probability and Statistics Using R*. Free Software Foundation, first edition, mar 2011.

[6] P. A. Macfarlane. Kansas geological survey, dakota aquifer program - ward's method, sep 1996.

[7] P. Ning and D. Xu. Learning attack strategies from intrusion alerts. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS '03, pages 200–209, New York, NY, USA, 2003. ACM.

[8] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu. Using of jaccard coefficient for keywords similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, page 6, 2013.

[9] R. Ruefle, A. Dorofee, D. Mundie, A. Householder, M. Murray, and S. Perl. Computer security incident response team development and evolution. *Security Privacy, IEEE*, 12(5):16–26, Sept 2014.

[10] A. Shameli-Sendi, R. Aghababaei-Barzegar, and M. Cheriet. Taxonomy of information security risk assessment (isra). *Computers & Security*, 57:14 – 30, 2016.

[11] R. Shittu, A. Healing, R. Ghanea-Hercock, R. Bloomfield, and M. Rajarajan. Intrusion alert prioritisation and attack detection using post-correlation analysis. *Computers & Security*, 50:1 – 15, 2015.

[12] G. P. Spathoulas and S. K. Katsikas. Enhancing ids performance through comprehensive alert post-processing. *Comput. Secur.*, 37:176–196, Sept. 2013.

[13] J. J. Treinen and R. Thurimella. A framework for the application of association rule mining in large intrusion detection infrastructures. In *Proceedings of the 9th International Conference on Recent Advances in Intrusion Detection*, RAID'06, pages 1–18, Berlin, Heidelberg, 2006. Springer-Verlag.

[14] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[15] R. Xu and I. Wunsch, D. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645–678, May 2005.

[16] F. Xuewei, W. Dongxia, H. Minhuan, and S. Xiaoxia. An approach of discovering causal knowledge for alert correlating based on data mining. In *Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on*, pages 57–62, Aug 2014.